CLAIMS

What is claimed is:

1. A computer system comprising:

a memory system comprising a plurality of memory cartridges, each of the plurality of memory cartridges comprising at least one memory device and a memory controller; and

a data controller comprising a plurality of control mechanisms, each of the plurality of control mechanisms corresponding to a respective one of the memory controllers and configured to independently interpret the transition of the corresponding memory cartridge between a first state of operation and a second state of operation, wherein the first state of operation permits the memory cartridge to be used to store data in a redundant memory array and wherein the second state of operation prevents the memory cartridge from being used to store data in a redundant memory array.

2. The computer system, as set forth in claim 1, wherein the at least one memory device comprises a dual inline memory module (DIMM).

3. The computer system, as set forth in claim 1, wherein the first state of operation comprises a redundant-ready state of operation.

4. The computer system, as set forth in claim 1, wherein the memory system is configured to operate in a redundant mode when each of the plurality of memory cartridges is in the redundant-ready state.

5. The computer system, as set forth in claim 1, wherein the second state of operation comprises one of a disable-up state, a disable-down state, a powerup state, a powerdown state, and a verify/replace state of operation.

6. The computer system, as set forth in claim 1, wherein the plurality of memory cartridges comprises five memory cartridges.

7. The computer system, as set forth in claim 1, wherein at least one of the plurality of memory cartridges is configured to store parity data.

8. The computer system, as set forth in claim 1, wherein each memory controller is configured to control access to the at least one memory device on the corresponding memory cartridge.

9. The computer system, as set forth in claim 1, wherein the data controller writes data in a striped fashion across the plurality of memory cartridges.

10. The computer system, as set forth in claim 1, wherein each of the plurality of control mechanisms is configured to independently facilitate the transition of the corresponding memory cartridge to one of a redundant-ready state, a powerdown state, a powerup state, a disable-down state, a disable-up state, and a verify/replace state.

11. A method of operating a memory system in a redundant mode, the memory system comprising a plurality of memory cartridges, comprising the act of independently transitioning each of the plurality of memory cartridges to a redundant-ready state.

12. A method of hot-plugging a memory cartridge comprising the acts of:

(a) operating the memory cartridge in a redundant-ready state;

(b) transitioning the memory cartridge from the redundant-ready state to a disable-down state to facilitate a de-assertion sequence of control signals;

(c) transitioning the memory cartridge from the disable-down state to a powerdown state wherein power is de-asserted to the memory cartridge;

(d) replacing at least a portion of the memory cartridge;

(e) transitioning the memory cartridge from the powerdown state to a powerup state wherein power is asserted to the memory cartridge;

(f) transitioning the memory cartridge from the powerup state to a disable-up state to facilitate an assertion sequence of control signals;

(g) transitioning the memory cartridge from the disable-up state to a verify/replace state to verify the proper functionality of the memory cartridge and to replace data stored in the memory cartridge with valid system data; and

(h) transitioning the memory cartridge from the verify/replace state to the redundant-ready state.

13. The method, as set forth in claim 12, wherein act (a) comprises the act of utilizing error correction code to correct errors to prevent the occurrence of a fault.

14. The method, as set forth in claim 12, wherein act (c) comprises the act of driving all write interface outputs and all read interface inputs to a logical zero.

15. The method, as set forth in claim 12, wherein act (d) comprises the act of replacing at least one memory device on the memory cartridge.

16. The method, as set forth in claim 12, wherein act (d) comprises the act of replacing the memory cartridge with a different memory cartridge.

17. The method, as set forth in claim 12, wherein act (e) comprises the act of driving all write interfaces to an idle state and all read interface inputs to a tristate condition.

18. The method, as set forth in claim 12, wherein act (g) comprises the acts of:

writing data to the memory cartridge; and

reading the data from the memory cartridge.

19. The method of claim 18, wherein the acts are performed in the recited order.

20. A method of passing fault information to a data controller comprising the acts of:

(a) reading data from a segment of a memory system;

(b) checking the data for errors at a memory controller;

(c) encoding information corresponding to results of the error checking; and

(d) delivering the encoded information from the memory controller to a data controller.

21. The method of passing fault information, as set forth in claim 20, comprising the act of embedding the encoded information in a pre-existing system signal.

22. The method of passing fault information, as set forth in claim 20, comprising the act of embedding the encoded information in a standard assertion singal.

23. The method of passing fault information, as set forth in claim 22, wherein act (d) comprises the act of embedding the encoded information corresponding to the results in an RxFRAME# signal.

24. The method of passing fault information, as set forth in claim 23, wherein act (d) comprises the act of embedding the encoded information on a second system clock cycle after initiation of the RxFRAME# signal.

25. The method of claim 20, comprising the acts of:

checking the data for errors at the data controller;

comparing the encoded information corresponding to results of the error checking in the memory controller with the error checking results in the data controller; and

generating a comparison error if the results of the error detection in the memory controller are different from the results of the error detection in the data controller.

26. The method of passing fault information, as set forth in claim 25, comprising the act of if a compare error is generated, replacing the corresponding segment of memory.

27. The method of passing fault information, as set forth in claim 20, wherein act (a) comprises the act of reading data from a segment of a redundant array of memory modules.

28. The method of passing fault information, as set forth in claim 20, wherein act (b) comprises the act of checking the data for errors at a memory controller using an ECC algorithm.

29. The method of passing fault information, as set forth in claim 20, wherein act (c) comprises the act of assigning a two-bit code corresponding to results of the error checking and indicating a result correlative to one of good data, corrected single bit error, uncorrected single bit error, and multi-bit error.